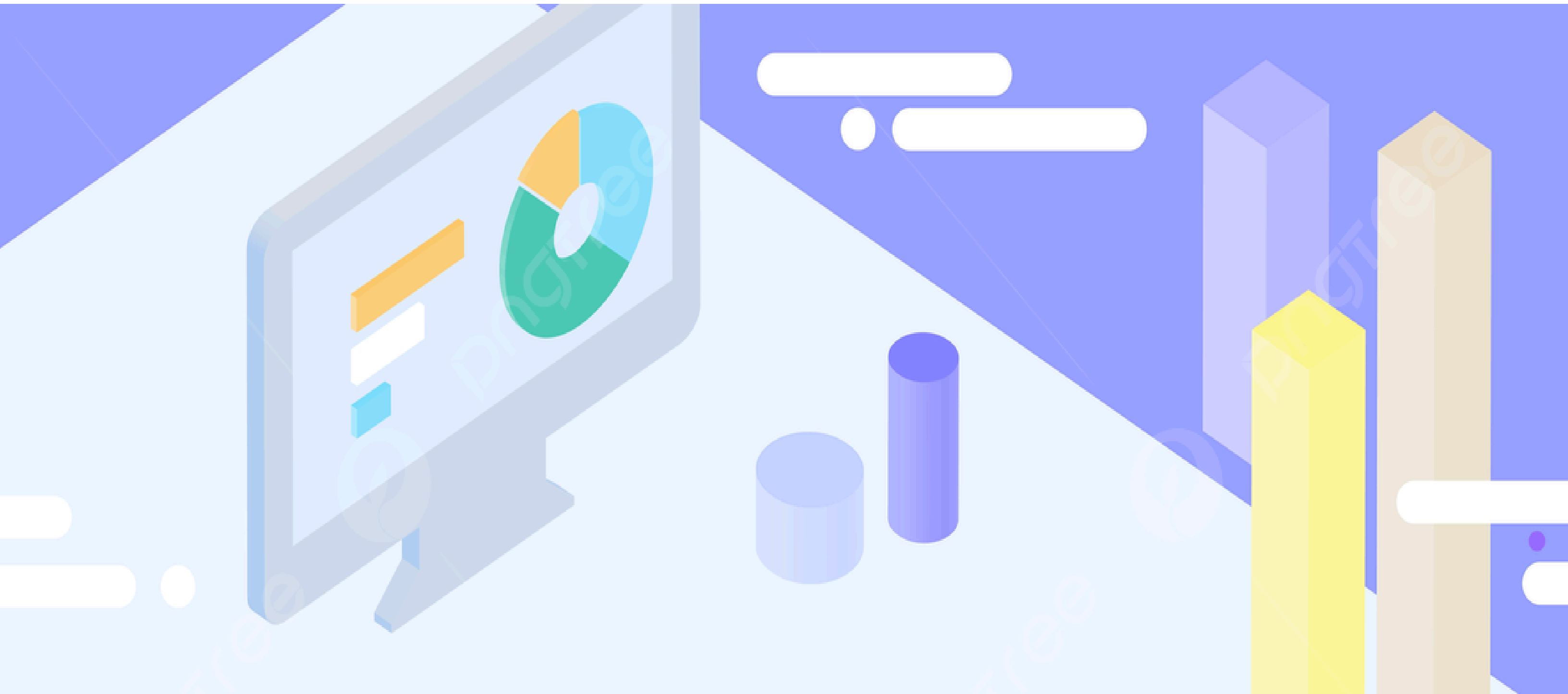
Data Visualization Individual Project

Name : Nada Youssef Ismail Youssef Amrawy

الاسم : ندى يوسف اسماعيل يوسف العمراوى

Id : 20221446068

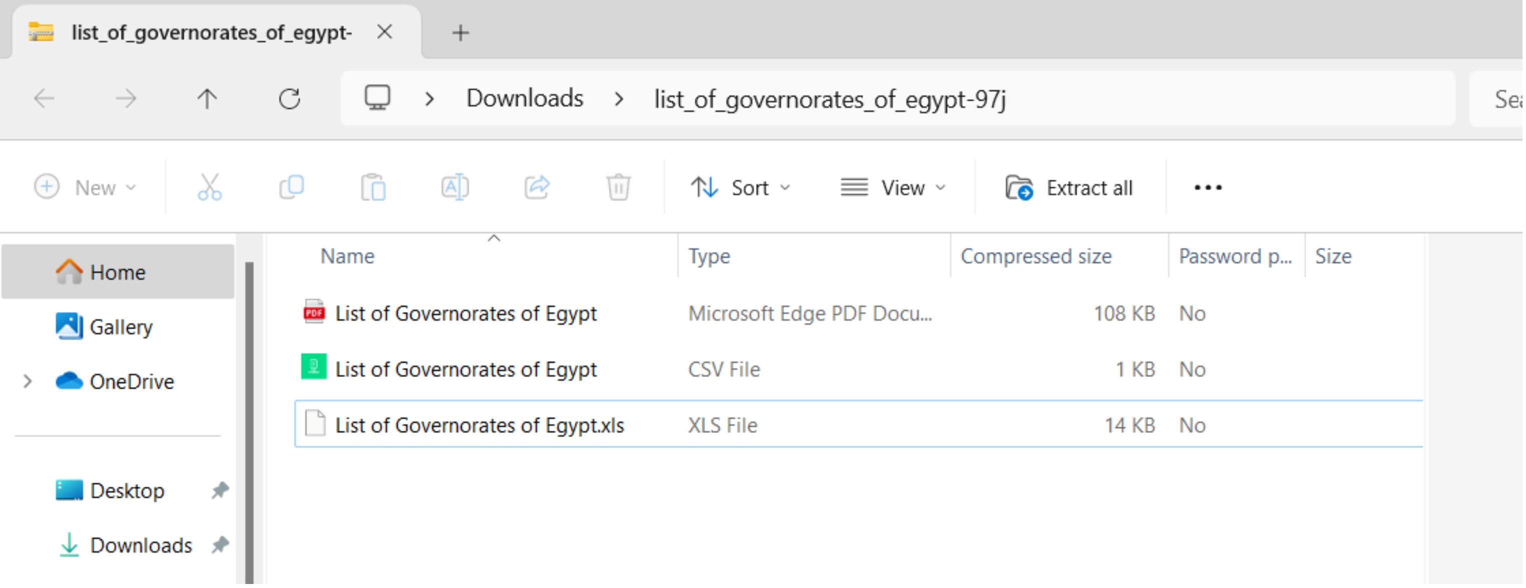


First Let’s Download a Dataset which is About Population Of 27 Cities In Egypt Government

- - - - X

I used the resource link from :downloadexcelfiles.com website which is an open source for downloading free datasets https://www.downloadexcelfiles.com/eg_en/download-excel-file-list-governorates-egypt#gsc.tab=0

This is how it looks like

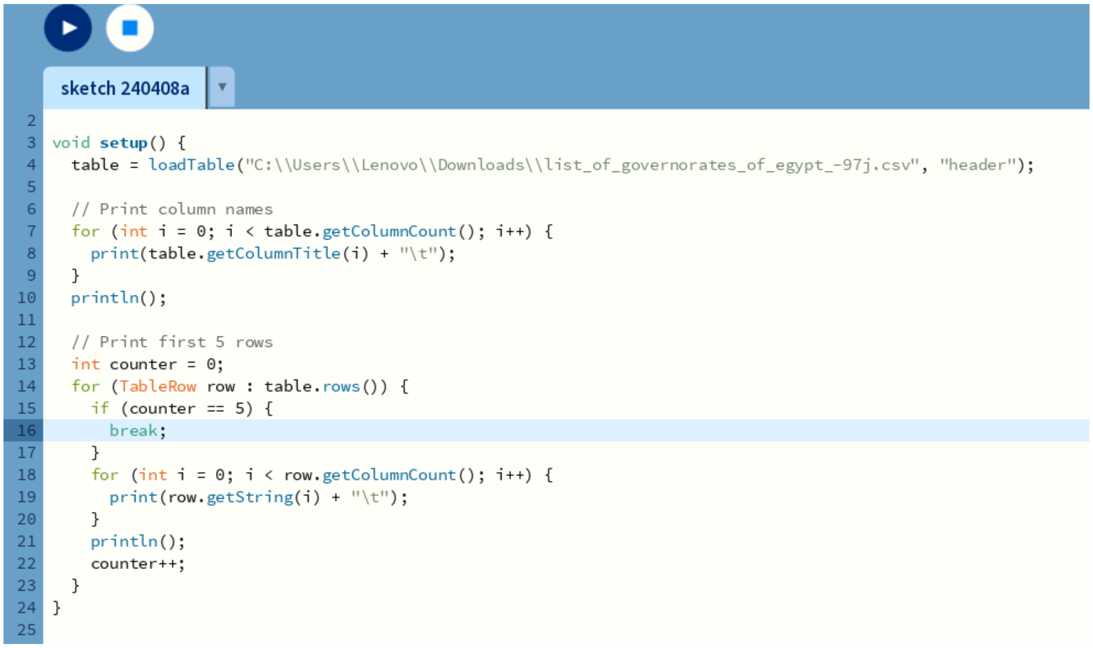

list_of_governorates_of_egypt-
Downloads > list_of_governorates_of_egypt-97j
Sea
New
Sort
View
Extract all
Home
Gallery
OneDrive
Desktop
Downloads

Name	Type	Compressed size	Password p...	Size
List of Governorates of Egypt	Microsoft Edge PDF Docu...	108 KB	No	
List of Governorates of Egypt	CSV File	1 KB	No	
List of Governorates of Egypt.xls	XLS File	14 KB	No	

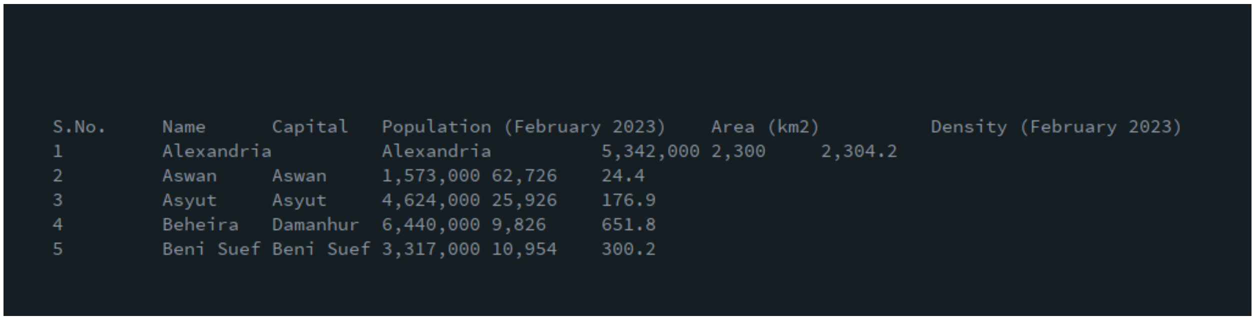
Step 2 : State the stages to visualize these data-Part 1 (Understanding, Planning & Preparing Data Phase)

- - - - X

A. Acquire the data and view first 5 rows in order to understand data



```
void setup() {
  table = loadTable("C:\\Users\\Lenovo\\Downloads\\list_of_governorates_of_egypt_-97j.csv", "header");

  // Print column names
  for (int i = 0; i < table.getColumnCount(); i++) {
    print(table.getColumnTitle(i) + "\t");
  }
  println();

  // Print first 5 rows
  int counter = 0;
  for (TableRow row : table.rows()) {
    if (counter == 5) {
      break;
    }
    for (int i = 0; i < row.getColumnCount(); i++) {
      print(row.getString(i) + "\t");
    }
    println();
    counter++;
  }
}

```



```
S.No.     Name      Capital   Population (February 2023)    Area (km2)          Density (February 2023)
1         Alexandria          Alexandria          5,342,000 2,300     2,304.2
2         Aswan     Aswan     1,573,000 62,726    24.4
3         Asyut     Asyut     4,624,000 25,926    176.9
4         Beheira   Damanhur  6,440,000 9,826     651.8
5         Beni Suef Beni Suef 3,317,000 10,954    300.2
```

So, what can we understand from here?

This dataset provides information about different governorates in Egypt. Each row represents a governorate and the columns provide various details about each governorate: Note : Last row is total governments calculations.

1. **S.No.**: This is the serial number of the governorate.
2. **Name**: This is the name of the governorate.
3. **Capital**: This is the capital city of the governorate.
4. **Population (February 2023)**: This is the population of the governorate as of February 2023.
5. **Area (km2)**: This is the area of the governorate in square kilometers.
6. **Density (February 2023)**: This is the population density of the governorate as of February 2023, calculated as the population divided by the area.

B. Ask questions that would lead us to meaningful visualization (I made a list of 5 questions)

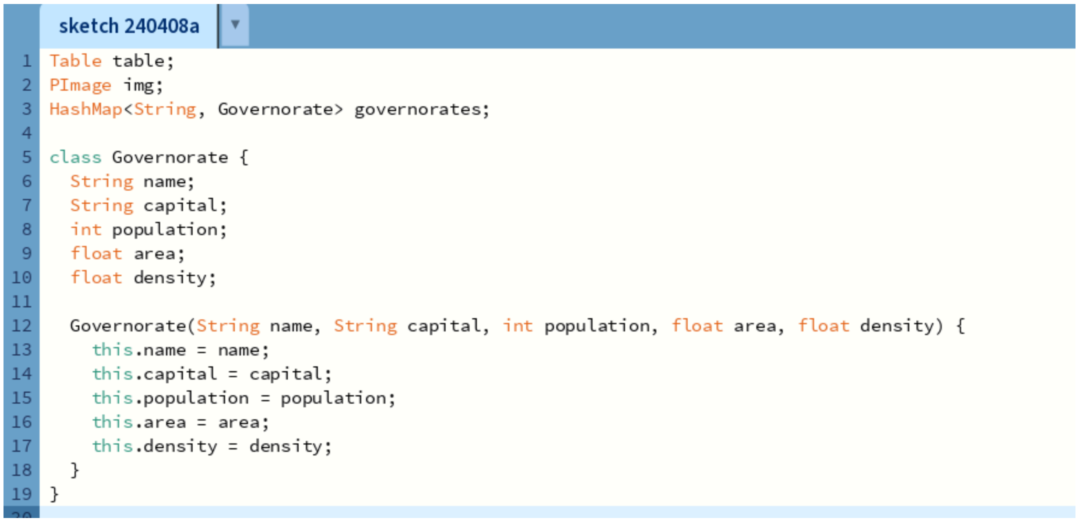
1. **Population Distribution**: How is the population distributed across the different governorates?
2. **Population Density**: Which governorates have the highest and lowest population densities?
3. **Area Comparison**: How do the areas of the governorates compare?
4. **Correlation of Population vs Area to Density** : How does the geographical area and total population of a governorate influence its population density, and what patterns or trends can be observed from these relationships?
5. **Geographical Visualization**: Can we plot the governorates on a map of Egypt and color-code them based on different metrics (population, area, density) and then let user interact to view my visual results?

C. Parse the data to a better format

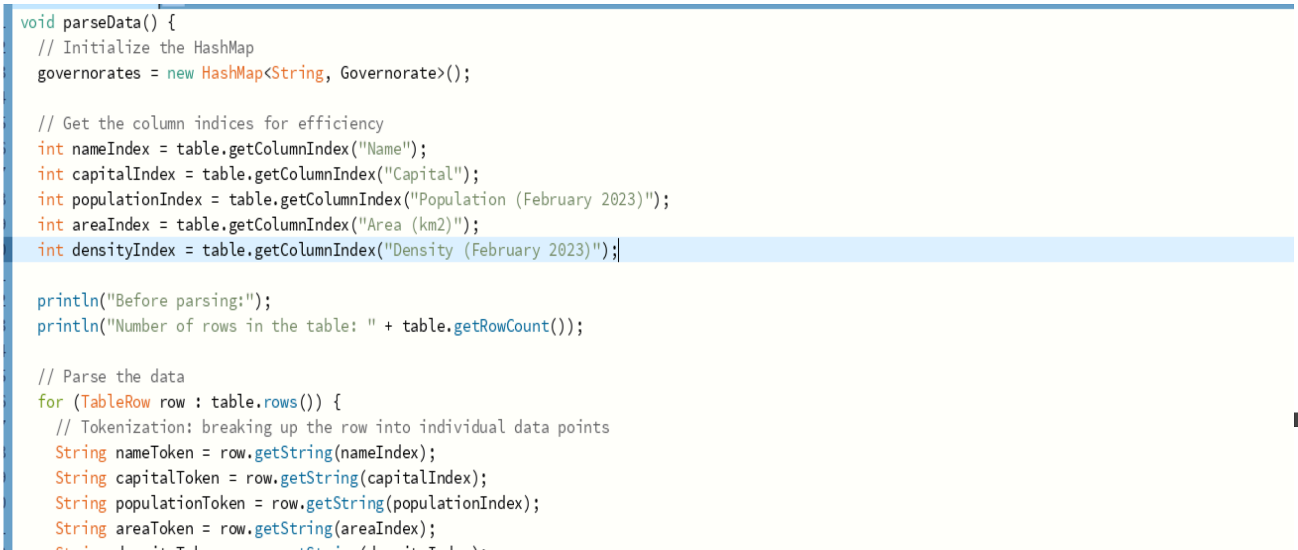
In the parsing stage, I implemented a function to transform the raw data of Egypt’s governorates into a structured format. I used four key techniques:

1. **Tokenization**: This breaks up each row into individual data points, allowing for the extraction of specific pieces of information.
2. **Identification**: This determines the type of each data point, such as whether it’s a string, integer, or float.
3. **Structuring**: This organizes the identified tokens into a more complex structure, specifically a `Governorate` object.
4. **Interpretation**: This stores the structured `Governorate` objects into a `HashMap` for efficient access and manipulation.

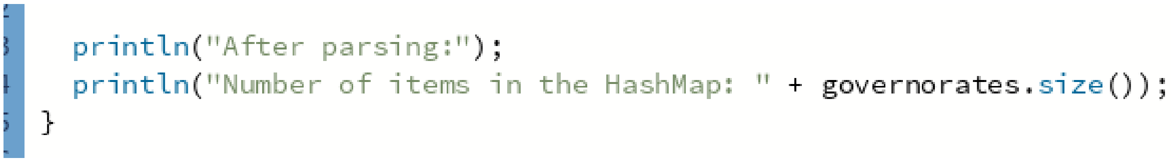
These techniques were used to transform the raw data into a more manageable and useful format, enabling efficient data manipulation for further processing or visualization. The effectiveness of the parsing was indicated by the number of items in the `HashMap` matching the number of rows in the table, barring any rows skipped due to missing or incomplete data.



```
Table table;
PImage img;
HashMap<String, Governorate> governorates;

class Governorate {
  String name;
  String capital;
  int population;
  float area;
  float density;

  Governorate(String name, String capital, int population, float area, float density) {
    this.name = name;
    this.capital = capital;
    this.population = population;
    this.area = area;
    this.density = density;
  }
}
```



```
void parseData() {
  // Initialize the HashMap
  governorates = new HashMap<String, Governorate>();

  // Get the column indices for efficiency
  int nameIndex = table.getColumnIndex("Name");
  int capitalIndex = table.getColumnIndex("Capital");
  int populationIndex = table.getColumnIndex("Population (February 2023)");
  int areaIndex = table.getColumnIndex("Area (km2)");
  int densityIndex = table.getColumnIndex("Density (February 2023)");

  println("Before parsing:");
  println("Number of rows in the table: " + table.getRowCount());

  // Parse the data
  for (TableRow row : table.rows()) {
    // Tokenization: breaking up the row into individual data points
    String nameToken = row.getString(nameIndex);
    String capitalToken = row.getString(capitalIndex);
    String populationToken = row.getString(populationIndex);
    String areaToken = row.getString(areaIndex);
```

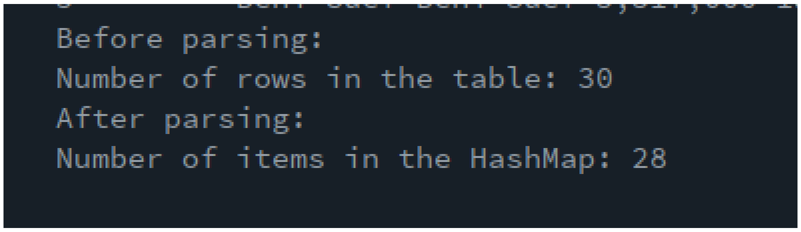


```
  println("After parsing:");
  println("Number of items in the HashMap: " + governorates.size());
}
```

And then I added the function parseData() to setup main function like this

 –>>> parseData();

So the output is



```
Before parsing:
Number of rows in the table: 30
After parsing:
Number of items in the HashMap: 28
```

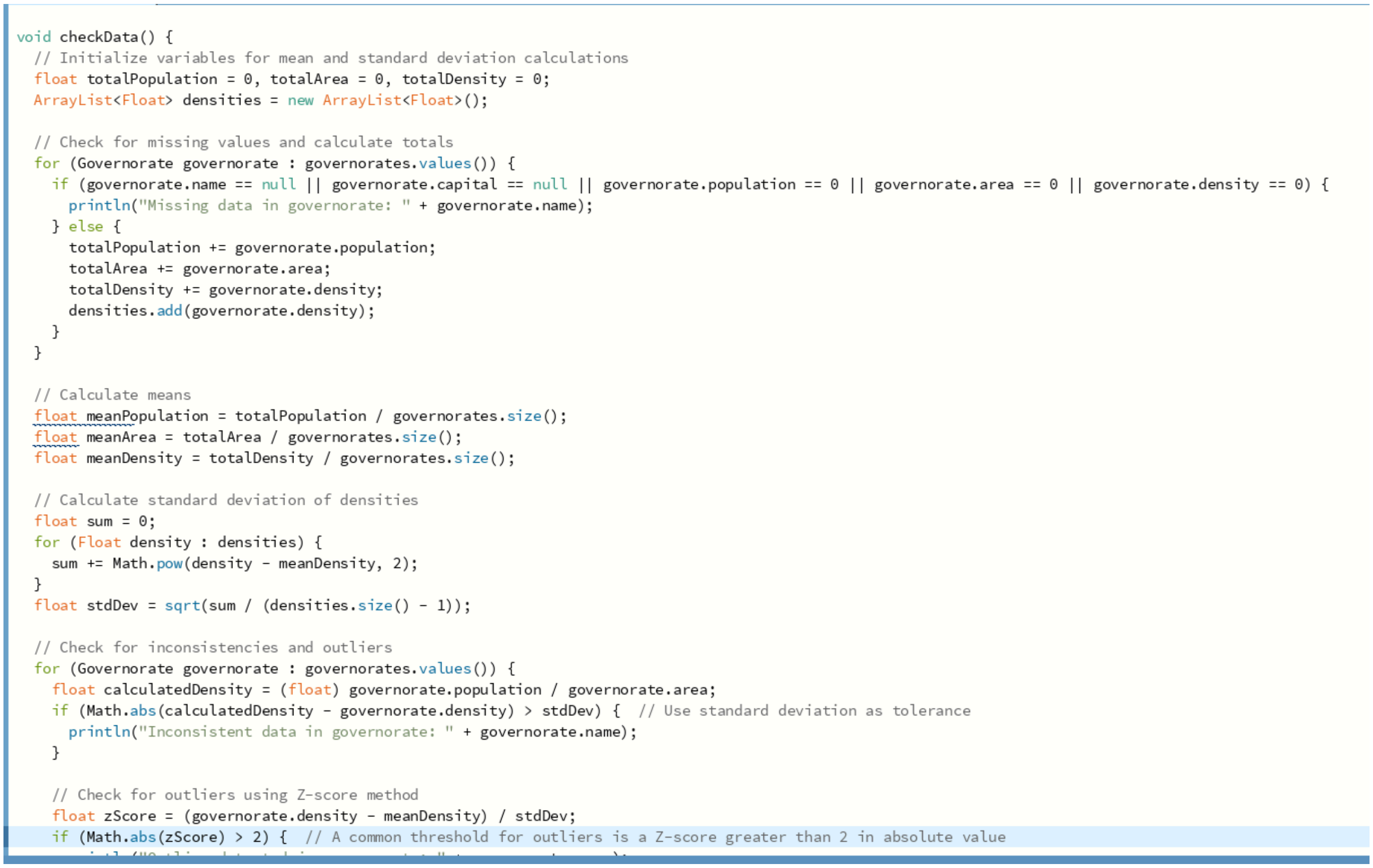
Step 3 : State the stages to visualize these data-Part 2 (Filtering & Mining)

A. Filtering

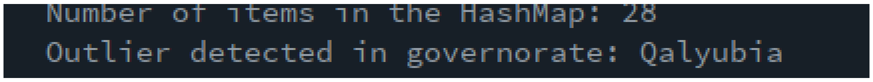
According to the nature of the dataset, which is a list of governorates with their respective population, area, and density, Here are some criteria to check if the dataset needs any form of filtering :

1. **Missing Values**: It's unlikely for a governorate to have a population, area, or density of zero. Therefore, checking for zero values is a reasonable way to detect missing data.
2. **Inconsistencies**: This check is based on the principle that the density of a governorate should be roughly equal to its population divided by its area. If the reported density is significantly different from the calculated density, it could indicate a data entry error or other inconsistency.
3. **Outliers**: a governorate with a population that is significantly higher or lower than the average might not necessarily be an outlier, but rather an indication of the diversity of the governorates. The Z-score method is used to detect outliers. This method is suitable for datasets with extreme values because it takes into account both the mean and the standard deviation.

So here I added a function to check if data needs to be filtered

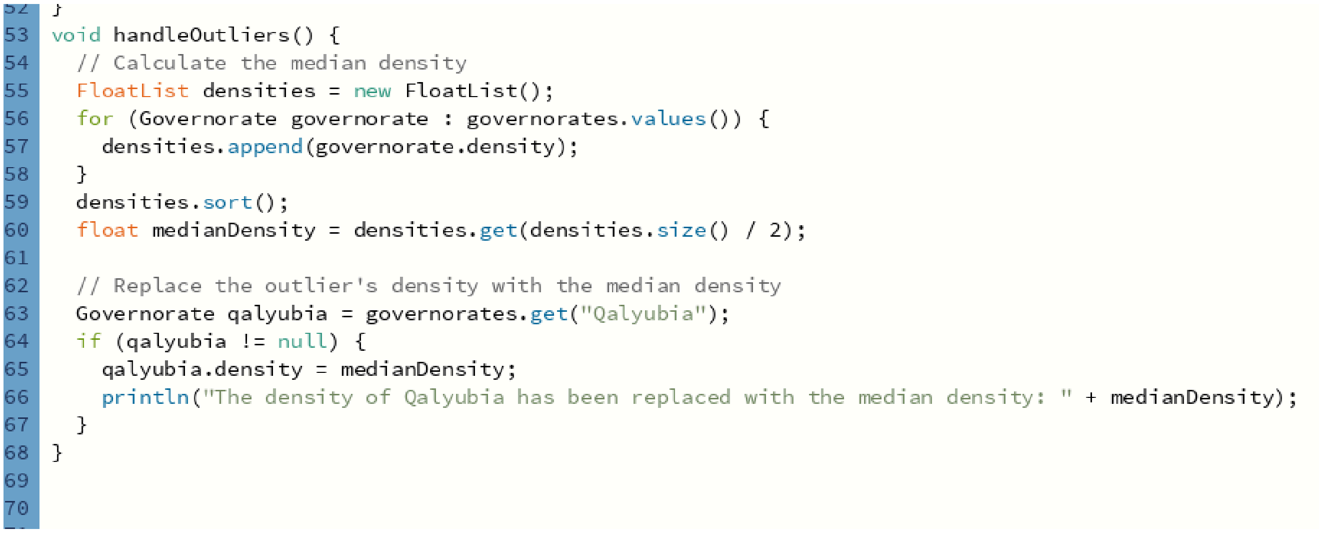


```
void checkData() {
  // Initialize variables for mean and standard deviation calculations
  float totalPopulation = 0, totalArea = 0, totalDensity = 0;
  ArrayList<Float> densities = new ArrayList<Float>();

  // Check for missing values and calculate totals
  for (Governorate governorate : governorates.values()) {
    if (governorate.name == null || governorate.capital == null || governorate.population == 0 || governorate.area == 0 || governorate.density == 0) {
      println("Missing data in governorate: " + governorate.name);
    } else {
      totalPopulation += governorate.population;
      totalArea += governorate.area;
      totalDensity += governorate.density;
      densities.add(governorate.density);
    }
  }

  // Calculate means
  float meanPopulation = totalPopulation / governorates.size();
  float meanArea = totalArea / governorates.size();
  float meanDensity = totalDensity / governorates.size();

  // Calculate standard deviation of densities
  float sum = 0;
  for (Float density : densities) {
    sum += Math.pow(density - meanDensity, 2);
  }
  float stdDev = sqrt(sum / (densities.size() - 1));

  // Check for inconsistencies and outliers
  for (Governorate governorate : governorates.values()) {
    float calculatedDensity = (float) governorate.population / governorate.area;
    if (Math.abs(calculatedDensity - governorate.density) > stdDev) {  // Use standard deviation as tolerance
      println("Inconsistent data in governorate: " + governorate.name);
    }

    // Check for outliers using Z-score method
    float zScore = (governorate.density - meanDensity) / stdDev;
    if (Math.abs(zScore) > 2) {  // A common threshold for outliers is a Z-score greater than 2 in absolute value
```

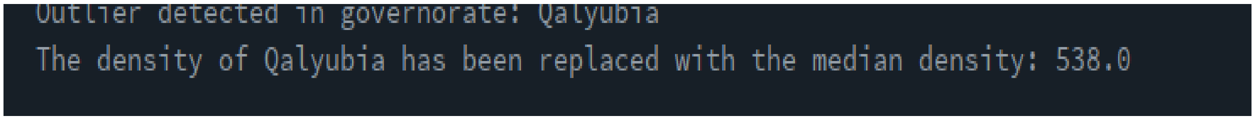


```
Number of items in the HashMap: 28
Outlier detected in governorate: Qalyubia
```

So the only issue is an outlier in Qalyubia..I decided to replace it with median value because this approach preserves the integrity of the dataset while reducing the impact of the extreme value on statistical calculations.



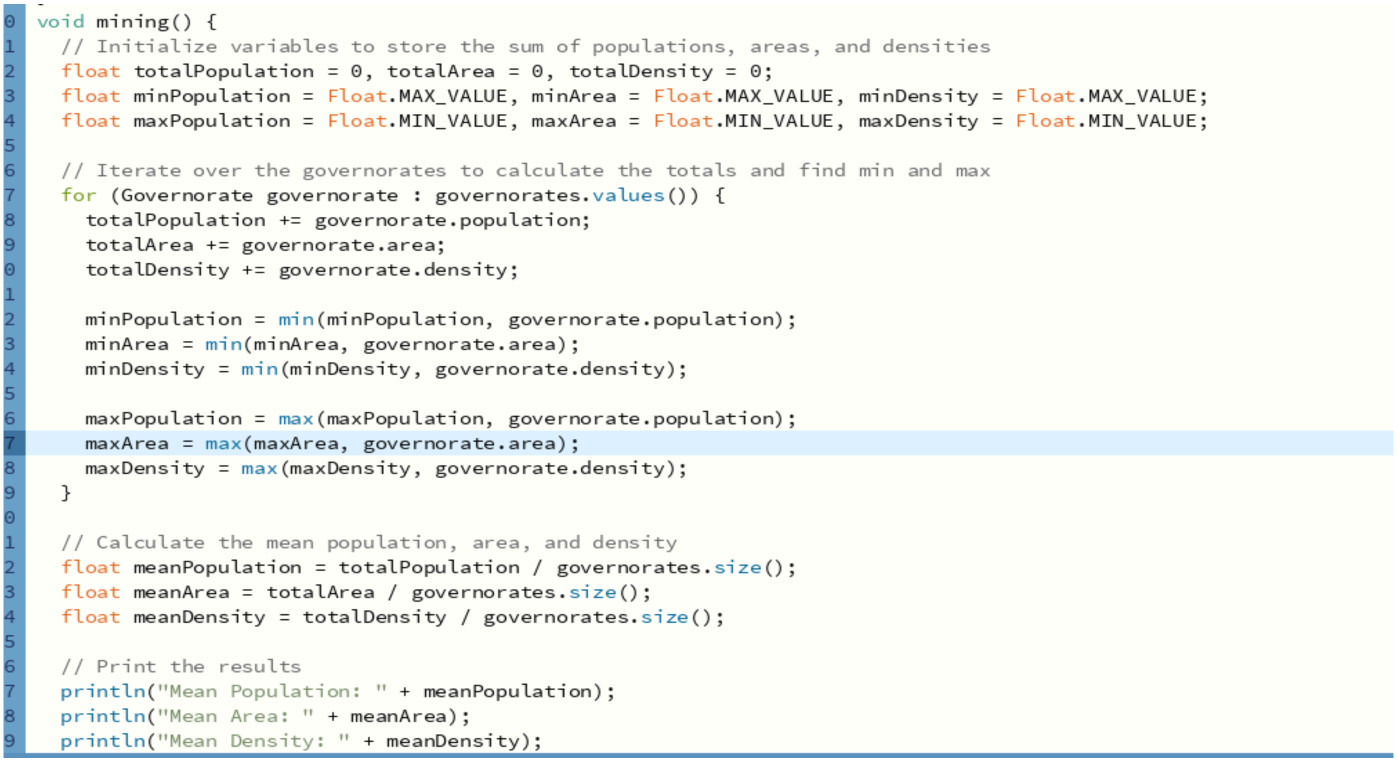
```
}
void handleOutliers() {
  // Calculate the median density
  FloatList densities = new FloatList();
  for (Governorate governorate : governorates.values()) {
    densities.append(governorate.density);
  }
  densities.sort();
  float medianDensity = densities.get(densities.size() / 2);

  // Replace the outlier's density with the median density
  Governorate qalyubia = governorates.get("Qalyubia");
  if (qalyubia != null) {
    qalyubia.density = medianDensity;
    println("The density of Qalyubia has been replaced with the median density: " + medianDensity);
  }
}
```



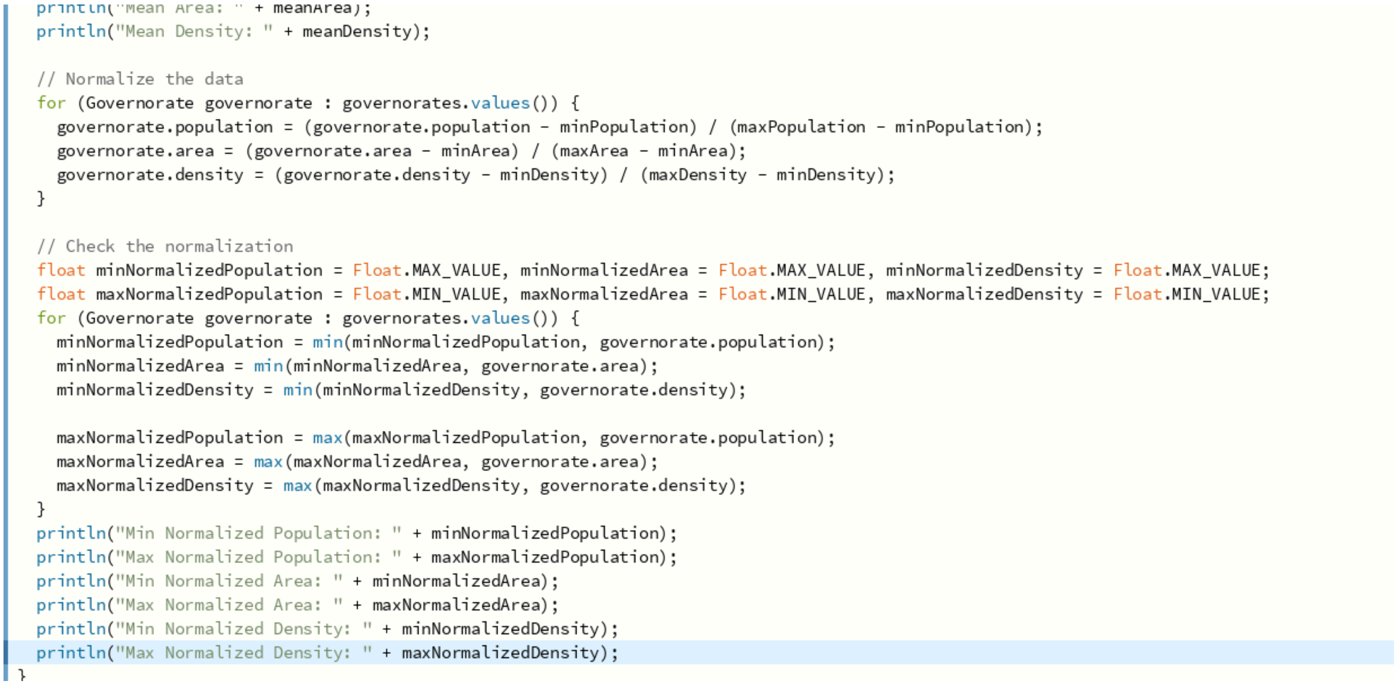
```
Outlier detected in governorate: Qalyubia
The density of Qalyubia has been replaced with the median density: 538.0
```

B. Mining

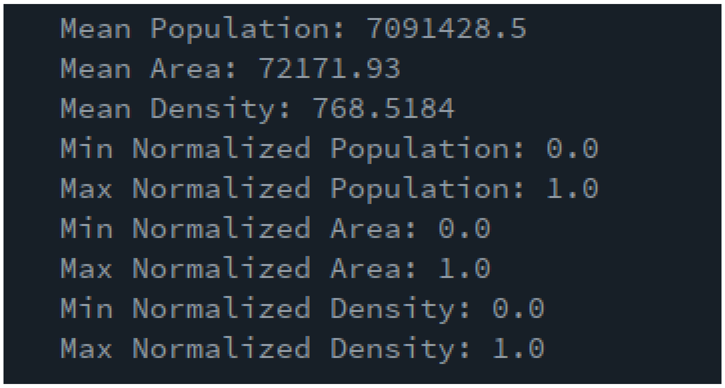
This stage is intended for a general data statistical understanding and scaling where i used mean & standard deviation as statistical description & normalization.The mean values represent the average population, area, and density across all governorates. The normalized values, which range from 0 to 1, allow for easier comparison between different governorates by putting them on the same scale.



```
void mining() {
  // Initialize variables to store the sum of populations, areas, and densities
  float totalPopulation = 0, totalArea = 0, totalDensity = 0;
  float minPopulation = Float.MAX_VALUE, minArea = Float.MAX_VALUE, minDensity = Float.MAX_VALUE;
  float maxPopulation = Float.MIN_VALUE, maxArea = Float.MIN_VALUE, maxDensity = Float.MIN_VALUE;

  // Iterate over the governorates to calculate the totals and find min and max
  for (Governorate governorate : governorates.values()) {
    totalPopulation += governorate.population;
    totalArea += governorate.area;
    totalDensity += governorate.density;

    minPopulation = min(minPopulation, governorate.population);
    minArea = min(minArea, governorate.area);
    minDensity = min(minDensity, governorate.density);

    maxPopulation = max(maxPopulation, governorate.population);
    maxArea = max(maxArea, governorate.area);
    maxDensity = max(maxDensity, governorate.density);
  }

  // Calculate the mean population, area, and density
  float meanPopulation = totalPopulation / governorates.size();
  float meanArea = totalArea / governorates.size();
  float meanDensity = totalDensity / governorates.size();

  // Print the results
  println("Mean Population: " + meanPopulation);
  println("Mean Area: " + meanArea);
  println("Mean Density: " + meanDensity);
```



```
  println("Mean Area: " + meanArea);
  println("Mean Density: " + meanDensity);

  // Normalize the data
  for (Governorate governorate : governorates.values()) {
    governorate.population = (governorate.population - minPopulation) / (maxPopulation - minPopulation);
    governorate.area = (governorate.area - minArea) / (maxArea - minArea);
    governorate.density = (governorate.density - minDensity) / (maxDensity - minDensity);
  }

  // Check the normalization
  float minNormalizedPopulation = Float.MAX_VALUE, minNormalizedArea = Float.MAX_VALUE, minNormalizedDensity = Float.MAX_VALUE;
  float maxNormalizedPopulation = Float.MIN_VALUE, maxNormalizedArea = Float.MIN_VALUE, maxNormalizedDensity = Float.MIN_VALUE;
  for (Governorate governorate : governorates.values()) {
    minNormalizedPopulation = min(minNormalizedPopulation, governorate.population);
    minNormalizedArea = min(minNormalizedArea, governorate.area);
    minNormalizedDensity = min(minNormalizedDensity, governorate.density);

    maxNormalizedPopulation = max(maxNormalizedPopulation, governorate.population);
    maxNormalizedArea = max(maxNormalizedArea, governorate.area);
    maxNormalizedDensity = max(maxNormalizedDensity, governorate.density);
  }
  println("Min Normalized Population: " + minNormalizedPopulation);
  println("Max Normalized Population: " + maxNormalizedPopulation);
  println("Min Normalized Area: " + minNormalizedArea);
  println("Max Normalized Area: " + maxNormalizedArea);
  println("Min Normalized Density: " + minNormalizedDensity);
  println("Max Normalized Density: " + maxNormalizedDensity);
}
```



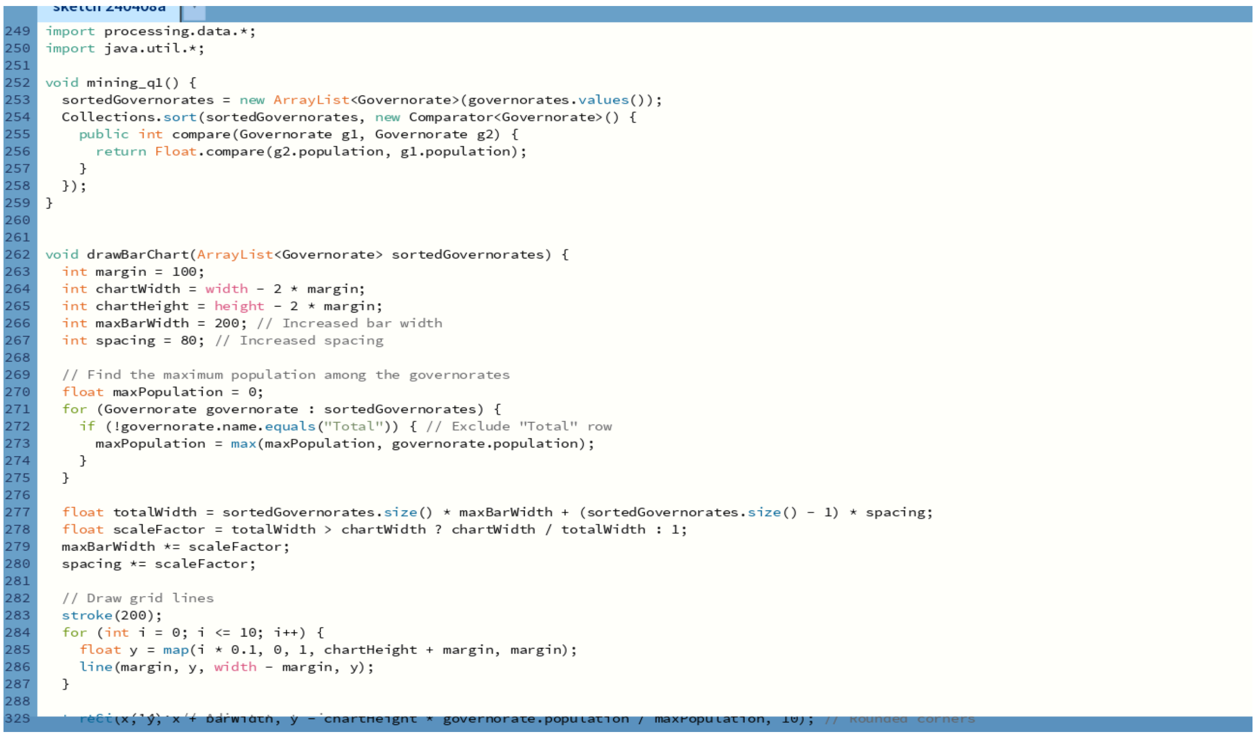
```
Mean Population: 7091428.5
Mean Area: 72171.93
Mean Density: 768.5184
Min Normalized Population: 0.0
Max Normalized Population: 1.0
Min Normalized Area: 0.0
Max Normalized Area: 1.0
Min Normalized Density: 0.0
Max Normalized Density: 1.0
```

Step 4 : State the stages to visualize these data-Part 3 (Specified Mining + Representing + Refine + Interact = Answer Questions Effectively)

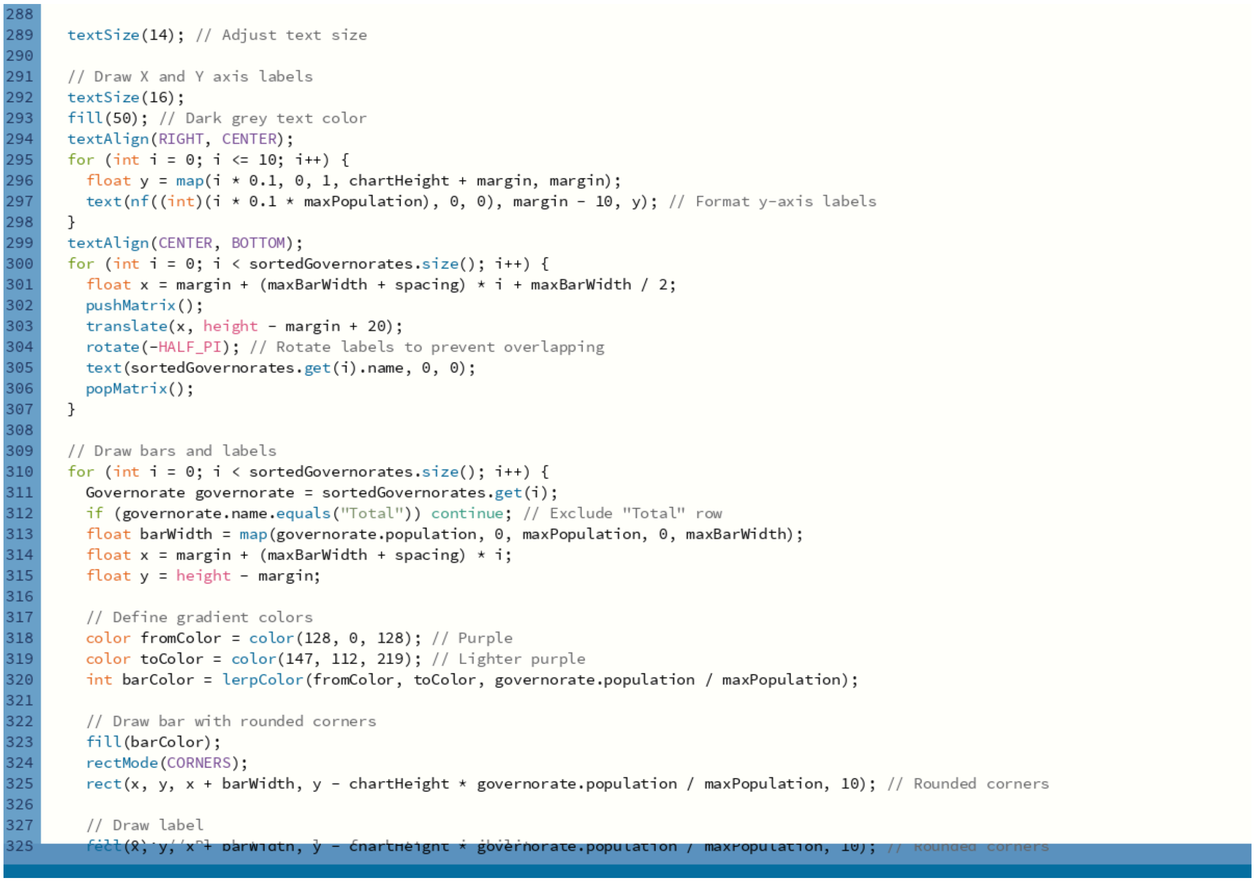
- - - - X

-Answering questions : extra data mining function for next visualization + Represent and refine at same time by constantly adjusting plots to look more defined and appealing to the user.
- At the end , will be using the map image to give user the ability to interact with it

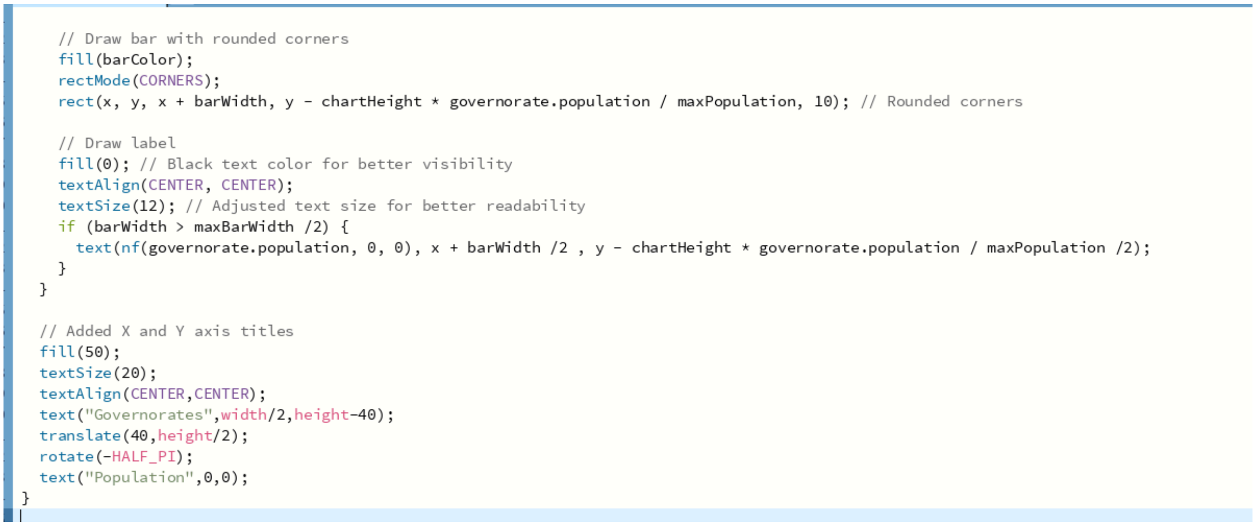
Q1Population Distribution: How is the population distributed across the different governorates?



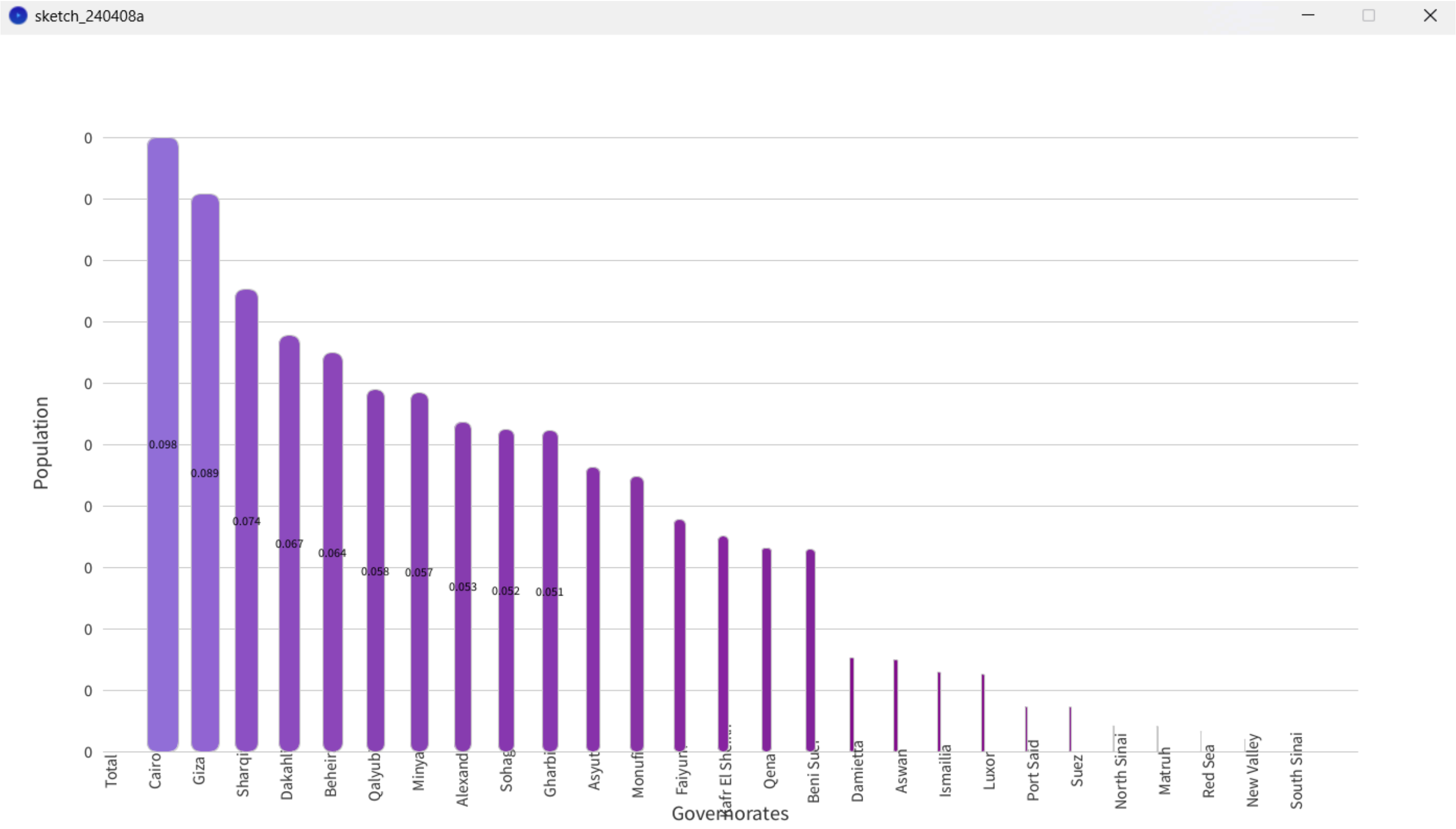
```
import processing.data.*;
import java.util.*;

void mining_q1() {
  sortedGovernorates = new ArrayList<Governorate>(governorates.values());
  Collections.sort(sortedGovernorates, new Comparator<Governorate>() {
    public int compare(Governorate g1, Governorate g2) {
      return Float.compare(g2.population, g1.population);
    }
  });
}


void drawBarChart(ArrayList<Governorate> sortedGovernorates) {
  int margin = 100;
  int chartWidth = width - 2 * margin;
  int chartHeight = height - 2 * margin;
  int maxBarWidth = 200; // Increased bar width
  int spacing = 80; // Increased spacing

  // Find the maximum population among the governorates
  float maxPopulation = 0;
  for (Governorate governorate : sortedGovernorates) {
    if (!governorate.name.equals("Total")) { // Exclude "Total" row
      maxPopulation = max(maxPopulation, governorate.population);
    }
  }

  float totalWidth = sortedGovernorates.size() * maxBarWidth + (sortedGovernorates.size() - 1) * spacing;
  float scaleFactor = totalWidth > chartWidth ? chartWidth / totalWidth : 1;
  maxBarWidth *= scaleFactor;
  spacing *= scaleFactor;

  // Draw grid lines
  stroke(200);
  for (int i = 0; i <= 10; i++) {
    float y = map(i * 0.1, 0, 1, chartHeight + margin, margin);
    line(margin, y, width - margin, y);
  }

    rect(x, y, x + barWidth, y - chartHeight * governorate.population / maxPopulation, 10); // Rounded corners
```



```

  textSize(14); // Adjust text size

  // Draw X and Y axis labels
  textSize(16);
  fill(50); // Dark grey text color
  textAlign(RIGHT, CENTER);
  for (int i = 0; i <= 10; i++) {
    float y = map(i * 0.1, 0, 1, chartHeight + margin, margin);
    text(nf((int)(i * 0.1 * maxPopulation), 0, 0), margin - 10, y); // Format y-axis labels
  }
  textAlign(CENTER, BOTTOM);
  for (int i = 0; i < sortedGovernorates.size(); i++) {
    float x = margin + (maxBarWidth + spacing) * i + maxBarWidth / 2;
    pushMatrix();
    translate(x, height - margin + 20);
    rotate(-HALF_PI); // Rotate labels to prevent overlapping
    text(sortedGovernorates.get(i).name, 0, 0);
    popMatrix();
  }

  // Draw bars and labels
  for (int i = 0; i < sortedGovernorates.size(); i++) {
    Governorate governorate = sortedGovernorates.get(i);
    if (governorate.name.equals("Total")) continue; // Exclude "Total" row
    float barWidth = map(governorate.population, 0, maxPopulation, 0, maxBarWidth);
    float x = margin + (maxBarWidth + spacing) * i;
    float y = height - margin;

    // Define gradient colors
    color fromColor = color(128, 0, 128); // Purple
    color toColor = color(147, 112, 219); // Lighter purple
    int barColor = lerpColor(fromColor, toColor, governorate.population / maxPopulation);

    // Draw bar with rounded corners
    fill(barColor);
    rectMode(CORNERS);
    rect(x, y, x + barWidth, y - chartHeight * governorate.population / maxPopulation, 10); // Rounded corners

    // Draw label
    fill(0); y, x + barWidth, y - chartHeight * governorate.population / maxPopulation, 10); // Rounded corners
```



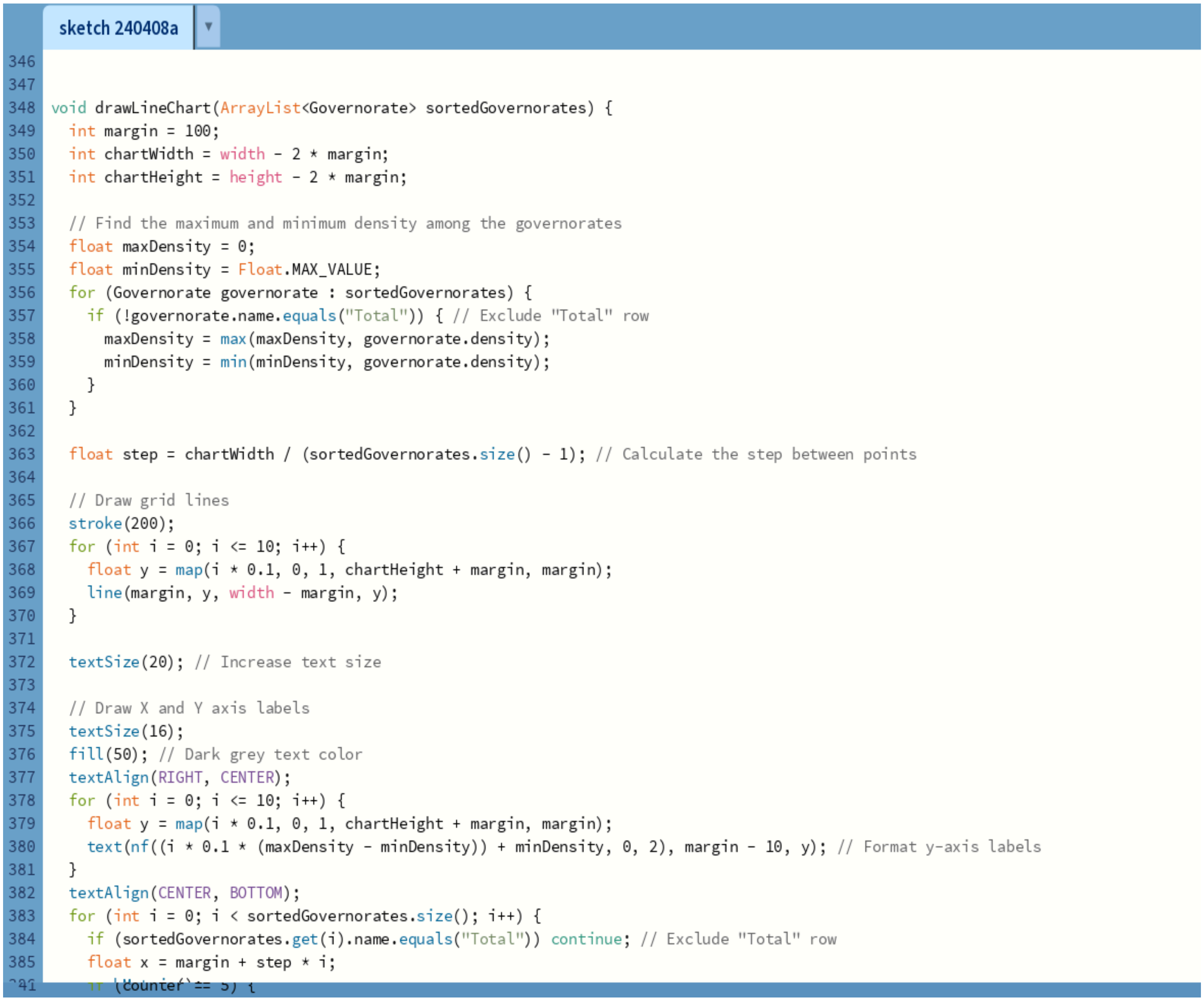
```
    // Draw bar with rounded corners
    fill(barColor);
    rectMode(CORNERS);
    rect(x, y, x + barWidth, y - chartHeight * governorate.population / maxPopulation, 10); // Rounded corners

    // Draw label
    fill(0); // Black text color for better visibility
    textAlign(CENTER, CENTER);
    textSize(12); // Adjusted text size for better readability
    if (barWidth > maxBarWidth /2) {
      text(nf(governorate.population, 0, 0), x + barWidth /2 , y - chartHeight * governorate.population / maxPopulation /2);
    }
  }

  // Added X and Y axis titles
  fill(50);
  textSize(20);
  textAlign(CENTER,CENTER);
  text("Governorates",width/2,height-40);
  translate(40,height/2);
  rotate(-HALF_PI);
  text("Population",0,0);
}
```



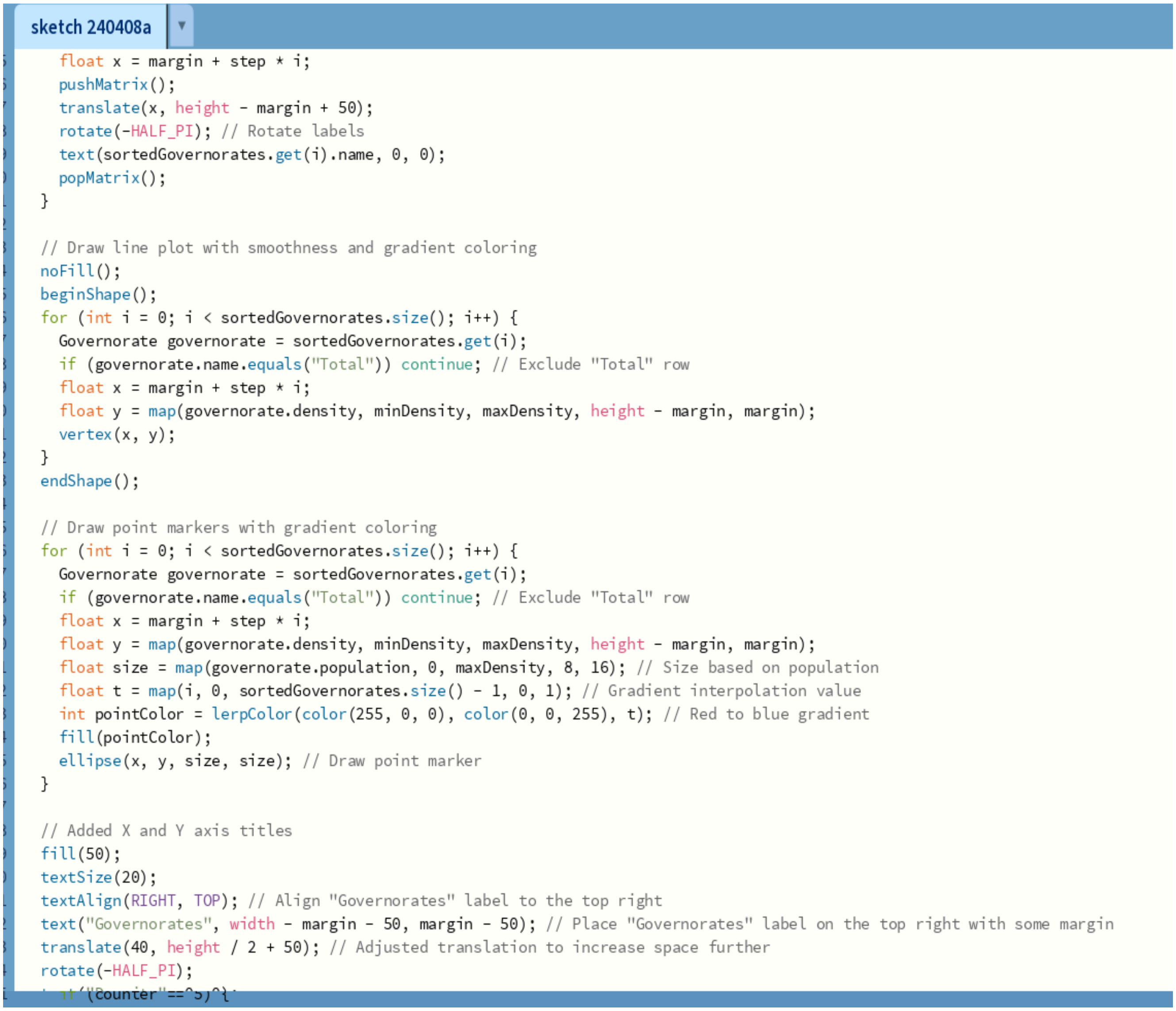
What does that tells us 👍

- Cairo & Giza have highest population
- Starting from Sohag to South Sinai are significantly lower
- The population distribution across different governorates as shown in the bar graph indicates a significant disparity. This uneven distribution could lead to various challenges e.g Resource Allocation for areas with highest population,economic disparity comparing areas of higher population vs areas with lower, Social Challenges like job hunting issues & overcrowded areas
- To address the issues arising from uneven population distribution, several strategies can be considered. These include promoting balanced development in less populated areas, implementing effective urban planning in densely populated areas, and controlling population growth through education and services. Additionally, decentralizing government services and offices to less populated areas can stimulate economic activity and balance population distribution.

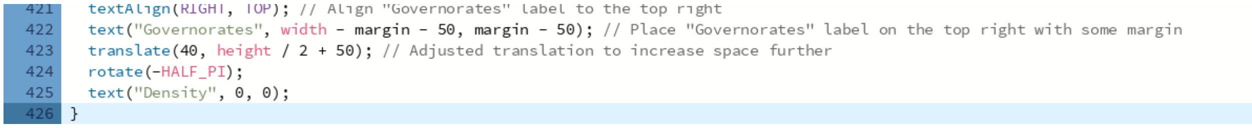
Q2 Population Density: Which governorates have the highest and lowest population densities?



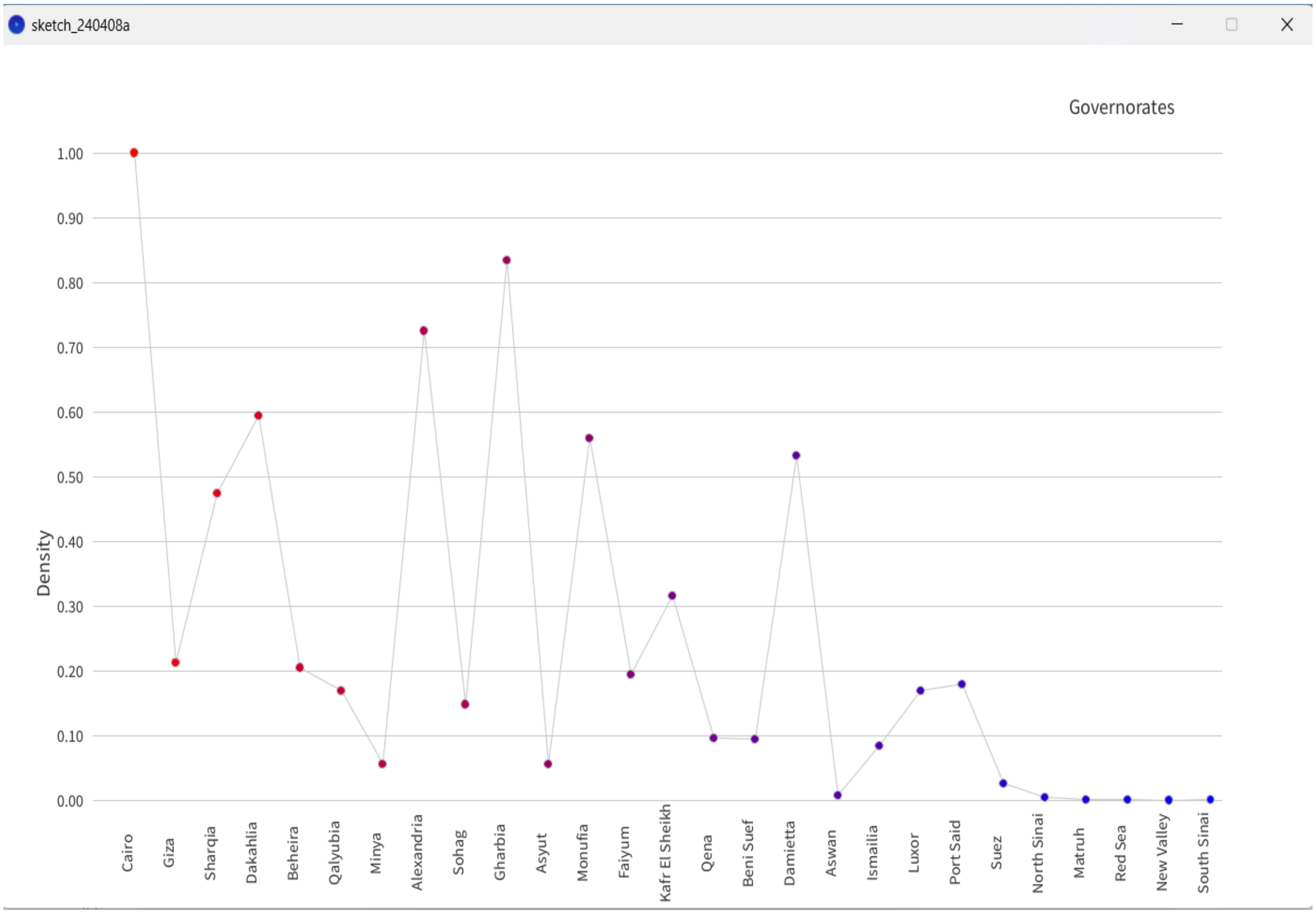
```
void drawLineChart(ArrayList<Governorate> sortedGovernorates) {
  int margin = 100;
  int chartWidth = width - 2 * margin;
  int chartHeight = height - 2 * margin;

  // Find the maximum and minimum density among the governorates
  float maxDensity = 0;
  float minDensity = Float.MAX_VALUE;
  for (Governorate governorate : sortedGovernorates) {
    if (!governorate.name.equals("Total")) { // Exclude "Total" row
      maxDensity = max(maxDensity, governorate.density);
      minDensity = min(minDensity, governorate.density);
    }
  }

  float step = chartWidth / (sortedGovernorates.size() - 1); // Calculate the step between points

  // Draw grid lines
  stroke(200);
  for (int i = 0; i <= 10; i++) {
    float y = map(i * 0.1, 0, 1, chartHeight + margin, margin);
    line(margin, y, width - margin, y);
  }

  textSize(20); // Increase text size

  // Draw X and Y axis labels
  textSize(16);
  fill(50); // Dark grey text color
  textAlign(RIGHT, CENTER);
  for (int i = 0; i <= 10; i++) {
    float y = map(i * 0.1, 0, 1, chartHeight + margin, margin);
    text(nf((i * 0.1 * (maxDensity - minDensity)) + minDensity, 0, 2), margin - 10, y); // Format y-axis labels
  }
  textAlign(CENTER, BOTTOM);
  for (int i = 0; i < sortedGovernorates.size(); i++) {
    if (sortedGovernorates.get(i).name.equals("Total")) continue; // Exclude "Total" row
    float x = margin + step * i;
    if (counter == 5) {
```



```
    float x = margin + step * i;
    pushMatrix();
    translate(x, height - margin + 50);
    rotate(-HALF_PI); // Rotate labels
    text(sortedGovernorates.get(i).name, 0, 0);
    popMatrix();
  }

  // Draw line plot with smoothness and gradient coloring
  noFill();
  beginShape();
  for (int i = 0; i < sortedGovernorates.size(); i++) {
    Governorate governorate = sortedGovernorates.get(i);
    if (governorate.name.equals("Total")) continue; // Exclude "Total" row
    float x = margin + step * i;
    float y = map(governorate.density, minDensity, maxDensity, height - margin, margin);
    vertex(x, y);
  }
  endShape();

  // Draw point markers with gradient coloring
  for (int i = 0; i < sortedGovernorates.size(); i++) {
    Governorate governorate = sortedGovernorates.get(i);
    if (governorate.name.equals("Total")) continue; // Exclude "Total" row
    float x = margin + step * i;
    float y = map(governorate.density, minDensity, maxDensity, height - margin, margin);
    float size = map(governorate.population, 0, maxDensity, 8, 16); // Size based on population
    float t = map(i, 0, sortedGovernorates.size() - 1, 0, 1); // Gradient interpolation value
    int pointColor = lerpColor(color(255, 0, 0), color(0, 0, 255), t); // Red to blue gradient
    fill(pointColor);
    ellipse(x, y, size, size); // Draw point marker
  }

  // Added X and Y axis titles
  fill(50);
  textSize(20);
  textAlign(RIGHT, TOP); // Align "Governorates" label to the top right
  text("Governorates", width - margin - 50, margin - 50); // Place "Governorates" label on the top right with some margin
  translate(40, height / 2 + 50); // Adjusted translation to increase space further
  rotate(-HALF_PI);
    if (counter==5){
```



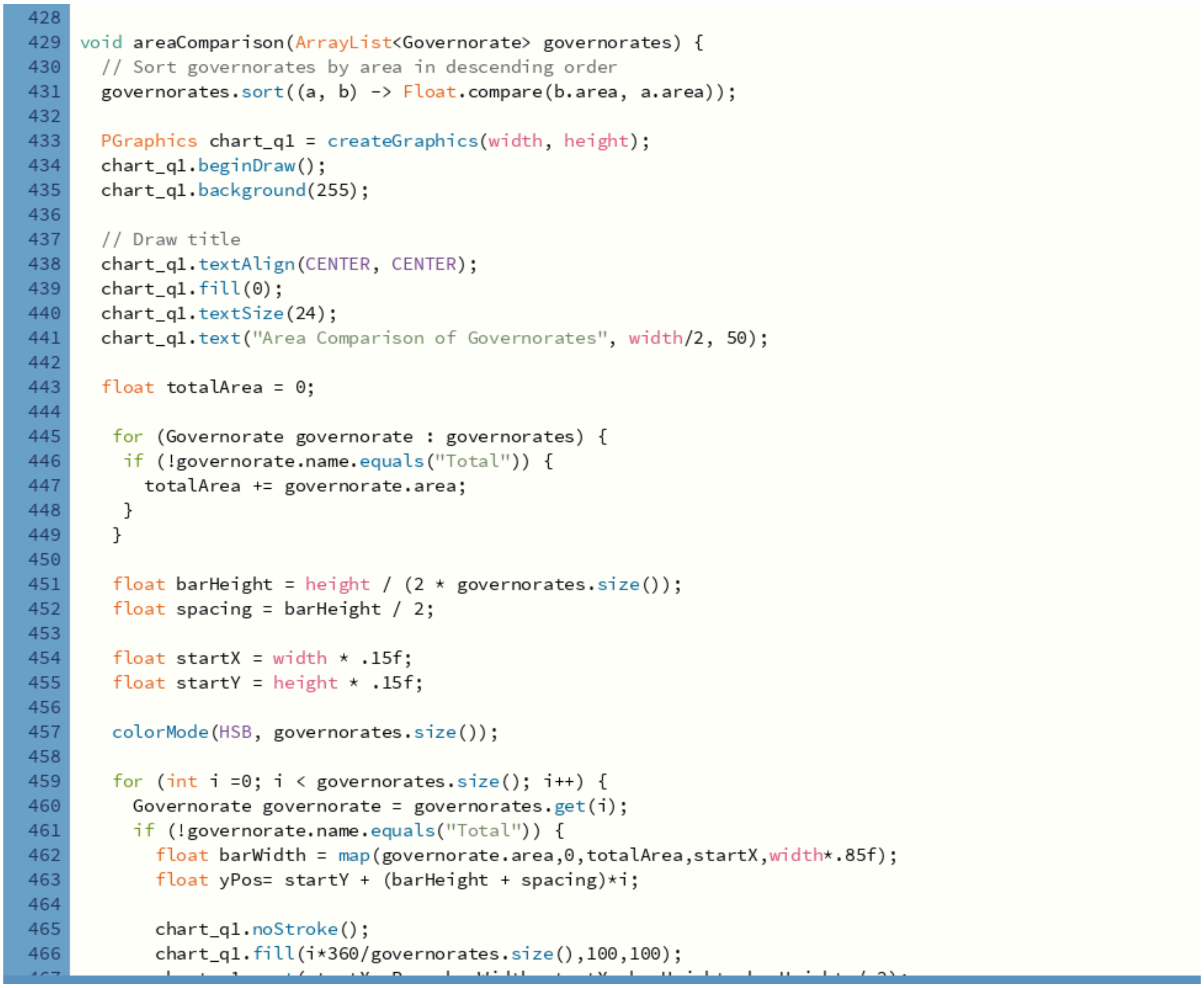
```
  textAlign(RIGHT, TOP); // Align "Governorates" label to the top right
  text("Governorates", width - margin - 50, margin - 50); // Place "Governorates" label on the top right with some margin
  translate(40, height / 2 + 50); // Adjusted translation to increase space further
  rotate(-HALF_PI);
  text("Density", 0, 0);
}
```



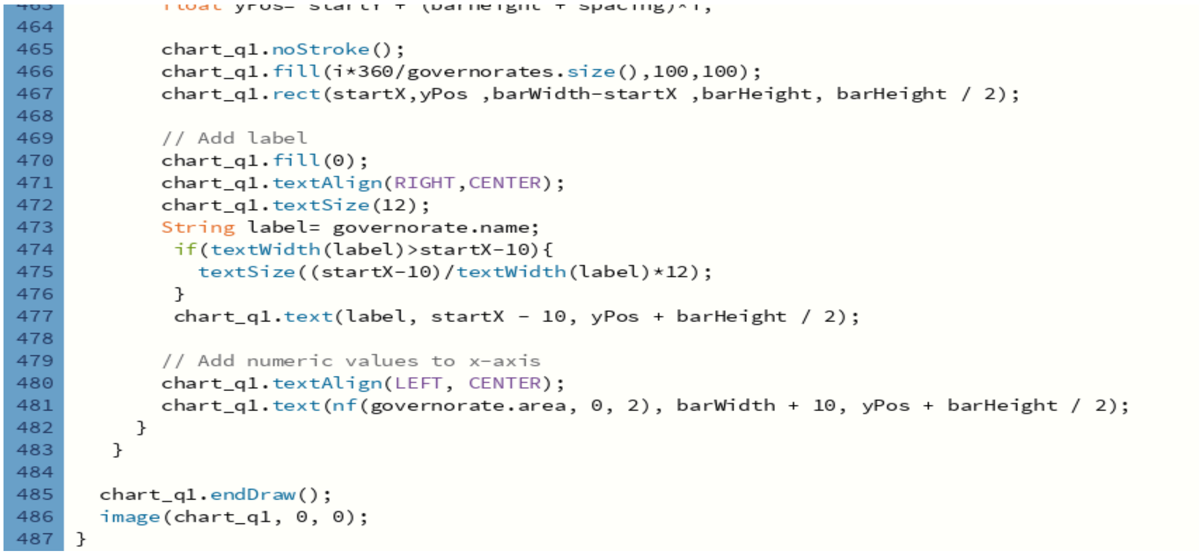
What does that tells us 👍

- The graph shows significant variations in population density among the governorates and a clear urban-rural divide. Some areas, like Cairo, have high density, indicating a large number of people living in a relatively small area. Others, like the New Valley, have much lower density, suggesting fewer people spread over a larger area.
- This information is crucial for urban planning and resource allocation. Areas with high population density might require more resources and services, such as public transportation, healthcare, and education. Conversely, areas with low population density might be underdeveloped and could be targeted for future growth and development.

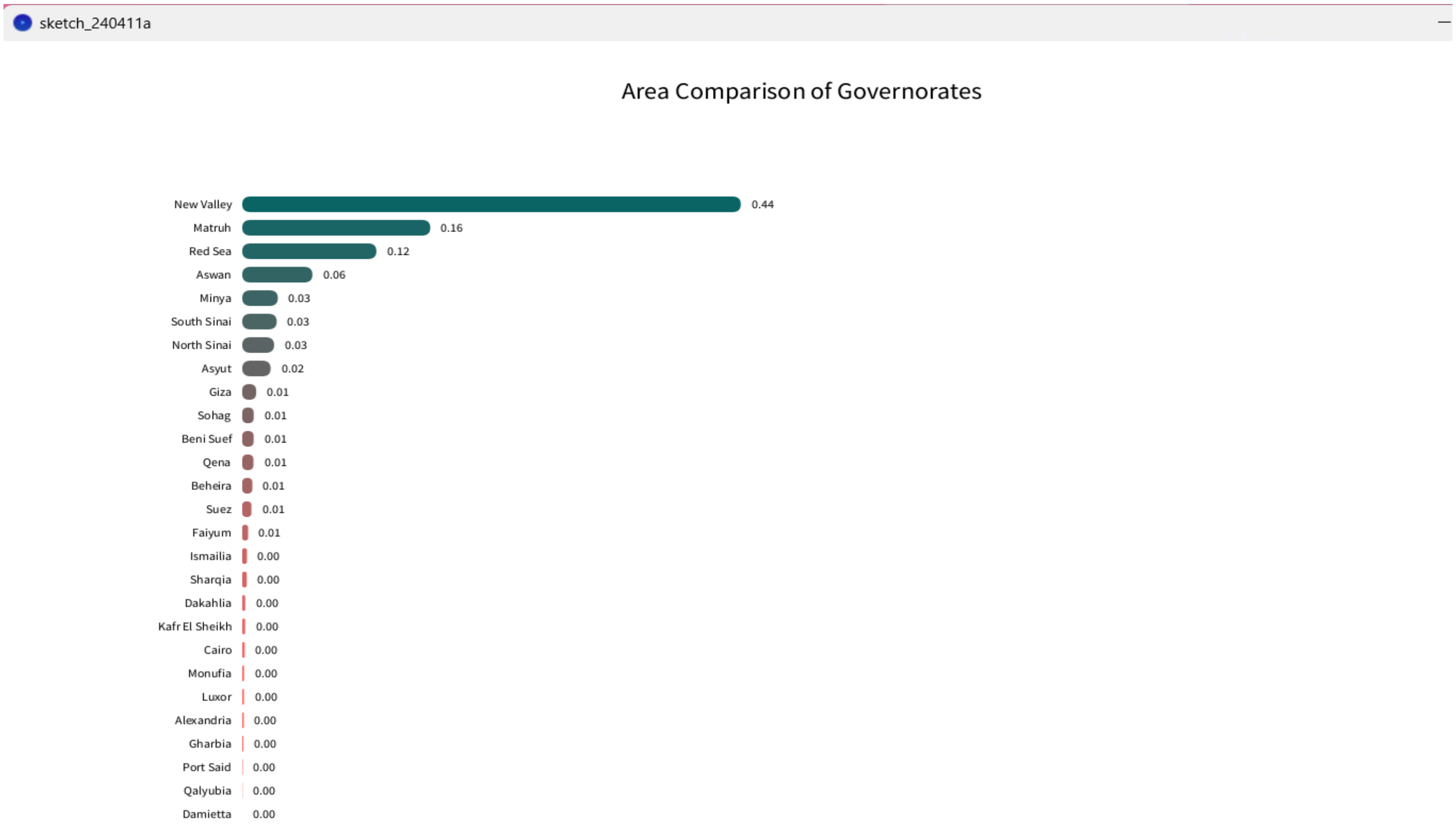
Q3 Area Comparison: How do the areas of the governorates compare?



```
void areaComparison(ArrayList<Governorate> governorates) {
  // Sort governorates by area in descending order
  governorates.sort((a, b) -> Float.compare(b.area, a.area));

  PGraphics chart_q1 = createGraphics(width, height);
  chart_q1.beginDraw();
  chart_q1.background(255);

  // Draw title
  chart_q1.textAlign(CENTER, CENTER);
  chart_q1.fill(0);
  chart_q1.textSize(24);
  chart_q1.text("Area Comparison of Governorates", width/2, 50);

  float totalArea = 0;

   for (Governorate governorate : governorates) {
    if (!governorate.name.equals("Total")) {
      totalArea += governorate.area;
    }
   }

   float barHeight = height / (2 * governorates.size());
   float spacing = barHeight / 2;

   float startX = width * .15f;
   float startY = height * .15f;

   colorMode(HSB, governorates.size());

   for (int i =0; i < governorates.size(); i++) {
     Governorate governorate = governorates.get(i);
     if (!governorate.name.equals("Total")) {
       float barWidth = map(governorate.area,0,totalArea,startX,width*.85f);
       float yPos= startY + (barHeight + spacing)*i;

       chart_q1.noStroke();
       chart_q1.fill(i*360/governorates.size(),100,100);
```



```
       float yPos= startY + (barHeight + spacing)*i;

       chart_q1.noStroke();
       chart_q1.fill(i*360/governorates.size(),100,100);
       chart_q1.rect(startX,yPos ,barWidth-startX ,barHeight, barHeight / 2);

       // Add label
       chart_q1.fill(0);
       chart_q1.textAlign(RIGHT,CENTER);
       chart_q1.textSize(12);
       String label= governorate.name;
        if(textWidth(label)>startX-10){
          textSize((startX-10)/textWidth(label)*12);
        }
        chart_q1.text(label, startX - 10, yPos + barHeight / 2);

       // Add numeric values to x-axis
       chart_q1.textAlign(LEFT, CENTER);
       chart_q1.text(nf(governorate.area, 0, 2), barWidth + 10, yPos + barHeight / 2);
     }
   }

  chart_q1.endDraw();
  image(chart_q1, 0, 0);
}
```



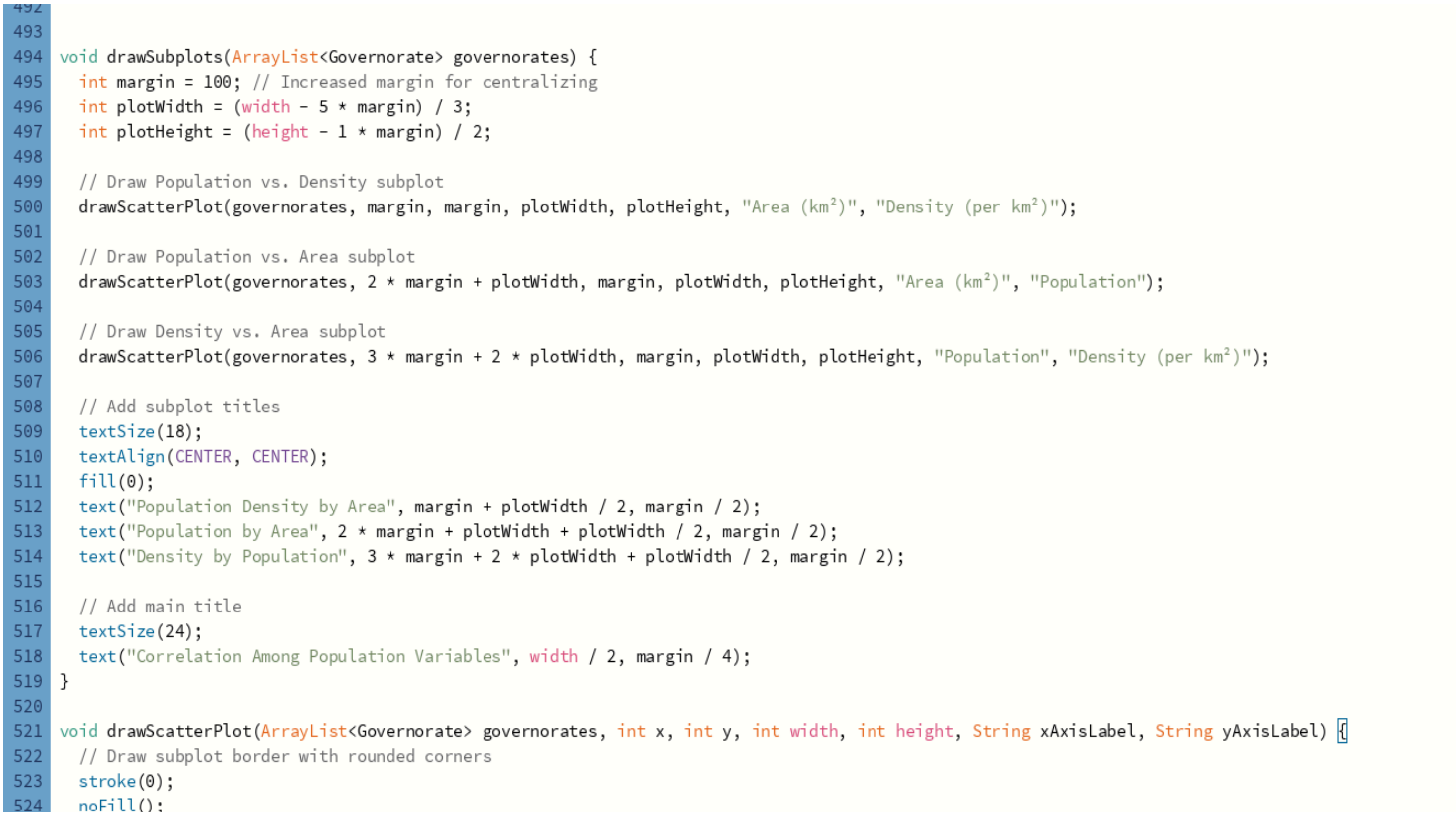
What does that tells us 👍

- New Valley has the largest area among the governorates, represented by the longest bar at 0.44.
- Matruh and Red Sea follow with areas represented by bars of length 0.16 and 0.12 respectively.
- Assiut and Minya have areas represented by bars of length 0.06 and 0.03 respectively.
- The remaining governorates, including South Sinai, North Sinai, Asyut, Qena, Sohag, Beni Suef, Sharqia, Suez, Faiyum, Ismailia, Beheira, Kafr El Sheikh, Giza, Cairo, Monufia, Luxor, Alexandria, Port Said, Qalyubia, and Damietta, have significantly smaller areas in comparison, with bars close to zero in length.

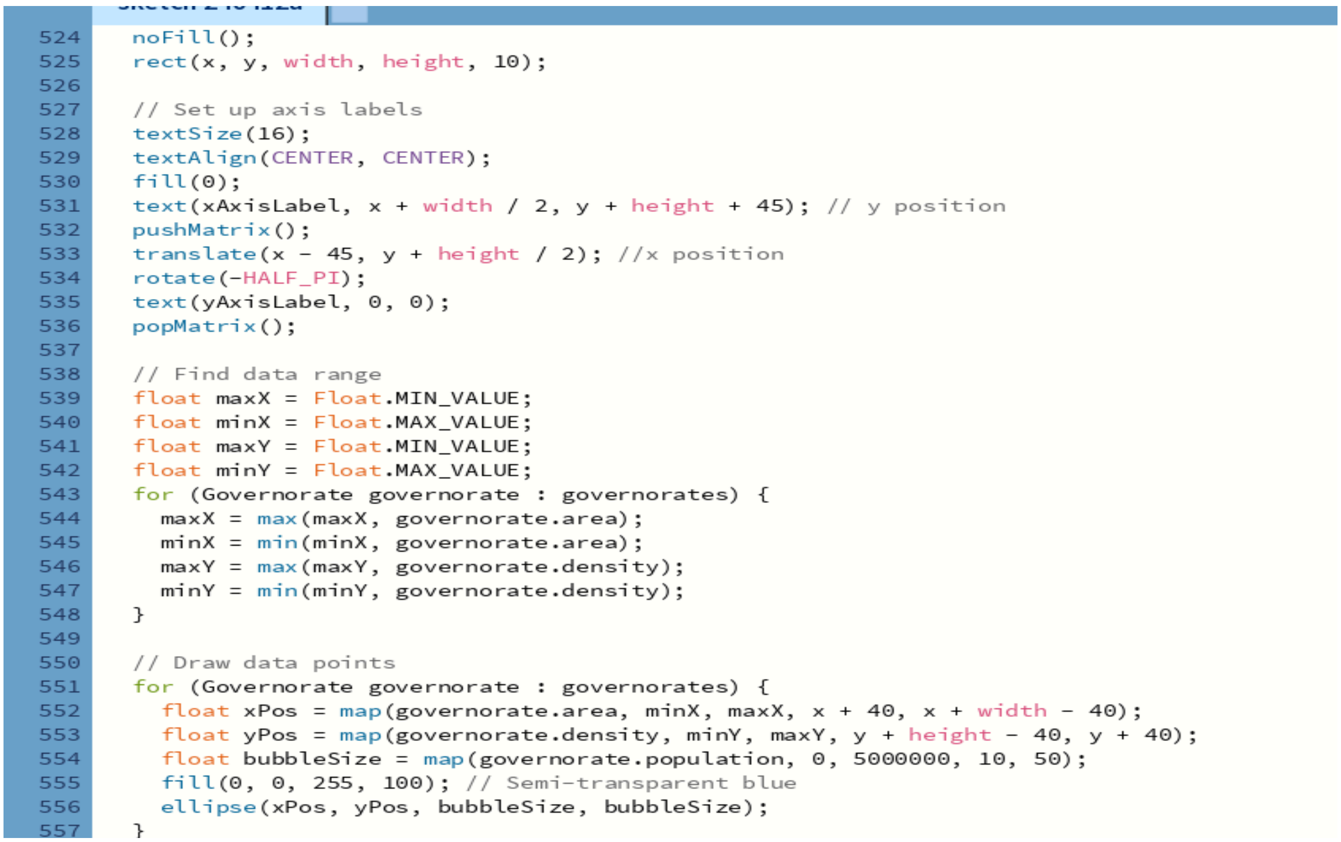
Q4 Correlation of Population vs Area to Density : How does the geographical area and total population of a governorate influence its population density, and what patterns or trends can be observed from these relationships?

- So here I’ve created 3 subplots in the canvas which are scatter plots that represent the correlation among :
  1. Population Density by Area
  2. Population by Area
  3. Density by Population

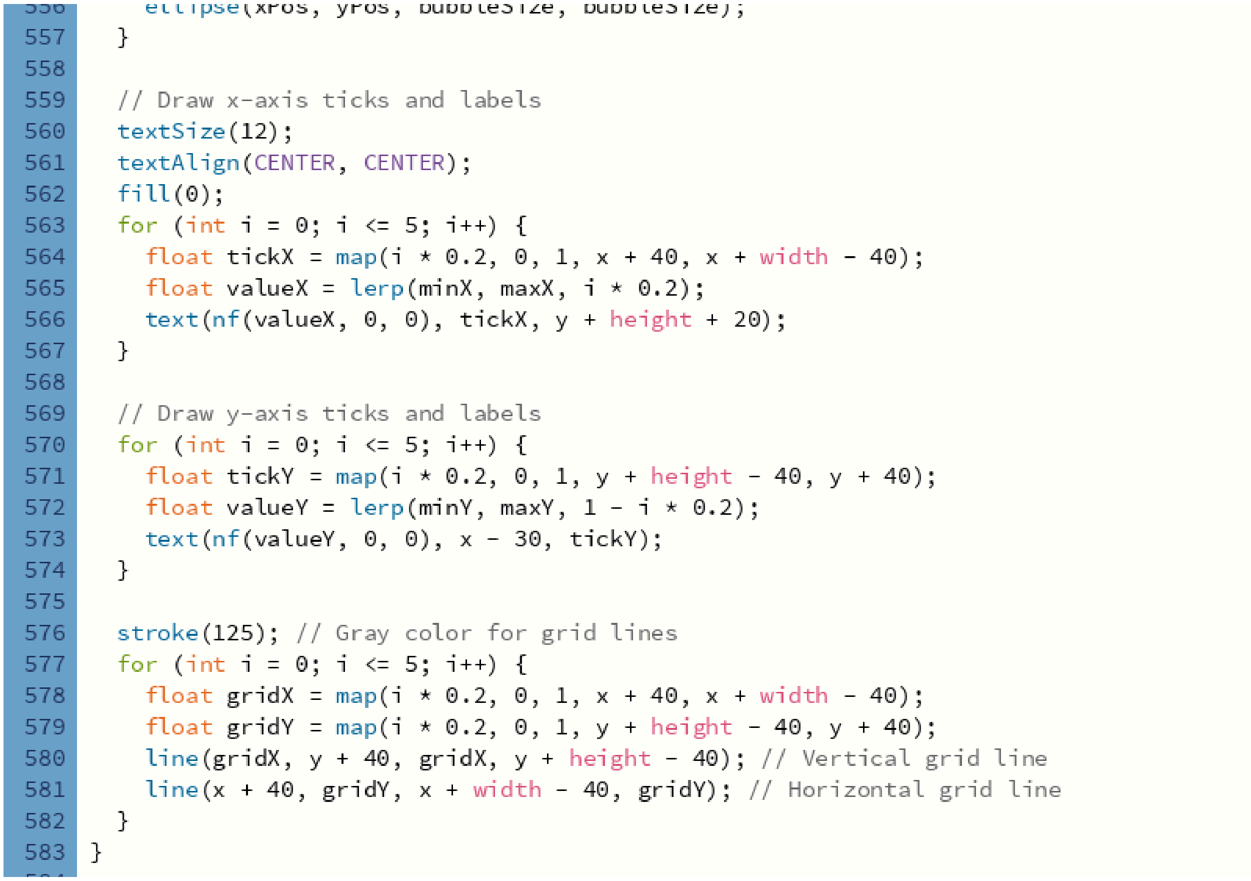
Check Code below



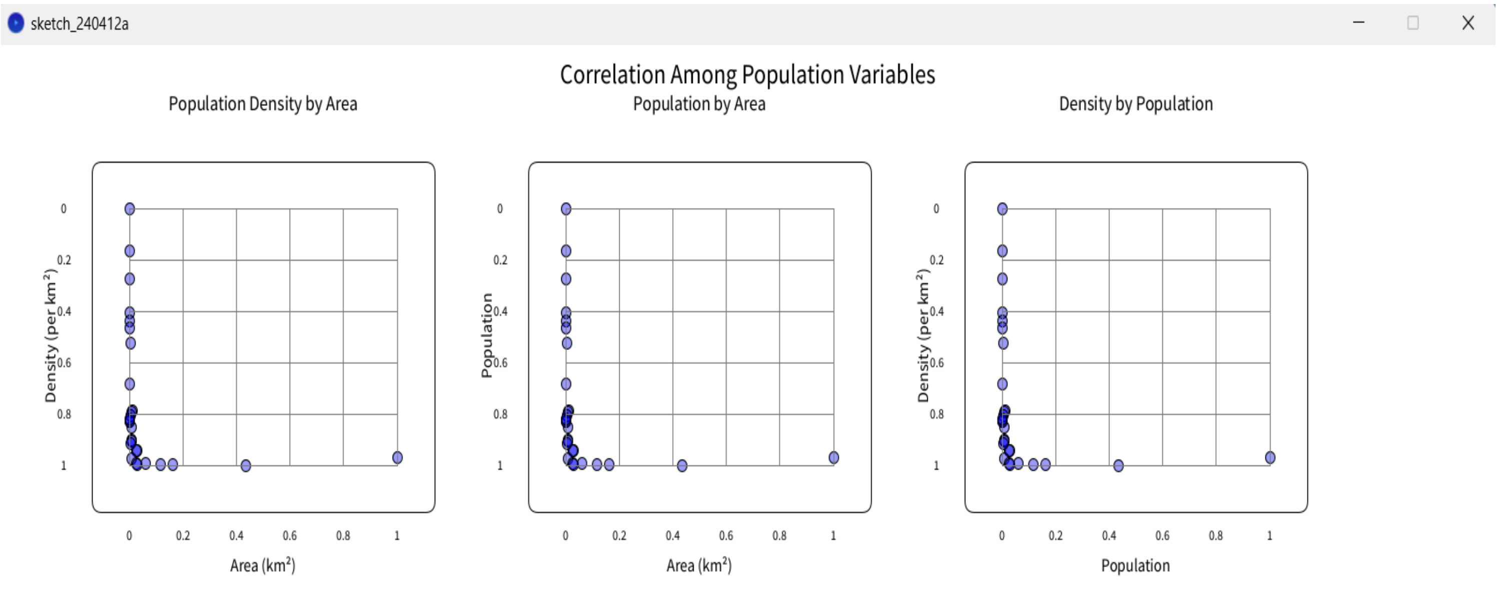
```

void drawSubplots(ArrayList<Governorate> governorates) {
  int margin = 100; // Increased margin for centralizing
  int plotWidth = (width - 5 * margin) / 3;
  int plotHeight = (height - 1 * margin) / 2;

  // Draw Population vs. Density subplot
  drawScatterPlot(governorates, margin, margin, plotWidth, plotHeight, "Area (km²)", "Density (per km²)");

  // Draw Population vs. Area subplot
  drawScatterPlot(governorates, 2 * margin + plotWidth, margin, plotWidth, plotHeight, "Area (km²)", "Population");

  // Draw Density vs. Area subplot
  drawScatterPlot(governorates, 3 * margin + 2 * plotWidth, margin, plotWidth, plotHeight, "Population", "Density (per km²)");

  // Add subplot titles
  textSize(18);
  textAlign(CENTER, CENTER);
  fill(0);
  text("Population Density by Area", margin + plotWidth / 2, margin / 2);
  text("Population by Area", 2 * margin + plotWidth + plotWidth / 2, margin / 2);
  text("Density by Population", 3 * margin + 2 * plotWidth + plotWidth / 2, margin / 2);

  // Add main title
  textSize(24);
  text("Correlation Among Population Variables", width / 2, margin / 4);
}

void drawScatterPlot(ArrayList<Governorate> governorates, int x, int y, int width, int height, String xAxisLabel, String yAxisLabel) {
  // Draw subplot border with rounded corners
  stroke(0);
  noFill();
```



```
  noFill();
  rect(x, y, width, height, 10);

  // Set up axis labels
  textSize(16);
  textAlign(CENTER, CENTER);
  fill(0);
  text(xAxisLabel, x + width / 2, y + height + 45); // y position
  pushMatrix();
  translate(x - 45, y + height / 2); //x position
  rotate(-HALF_PI);
  text(yAxisLabel, 0, 0);
  popMatrix();

  // Find data range
  float maxX = Float.MIN_VALUE;
  float minX = Float.MAX_VALUE;
  float maxY = Float.MIN_VALUE;
  float minY = Float.MAX_VALUE;
  for (Governorate governorate : governorates) {
    maxX = max(maxX, governorate.area);
    minX = min(minX, governorate.area);
    maxY = max(maxY, governorate.density);
    minY = min(minY, governorate.density);
  }

  // Draw data points
  for (Governorate governorate : governorates) {
    float xPos = map(governorate.area, minX, maxX, x + 40, x + width - 40);
    float yPos = map(governorate.density, minY, maxY, y + height - 40, y + 40);
    float bubbleSize = map(governorate.population, 0, 5000000, 10, 50);
    fill(0, 0, 255, 100); // Semi-transparent blue
    ellipse(xPos, yPos, bubbleSize, bubbleSize);
  }
```



```
    ellipse(xPos, yPos, bubbleSize, bubbleSize);
  }

  // Draw x-axis ticks and labels
  textSize(12);
  textAlign(CENTER, CENTER);
  fill(0);
  for (int i = 0; i <= 5; i++) {
    float tickX = map(i * 0.2, 0, 1, x + 40, x + width - 40);
    float valueX = lerp(minX, maxX, i * 0.2);
    text(nf(valueX, 0, 0), tickX, y + height + 20);
  }

  // Draw y-axis ticks and labels
  for (int i = 0; i <= 5; i++) {
    float tickY = map(i * 0.2, 0, 1, y + height - 40, y + 40);
    float valueY = lerp(minY, maxY, 1 - i * 0.2);
    text(nf(valueY, 0, 0), x - 30, tickY);
  }

  stroke(125); // Gray color for grid lines
  for (int i = 0; i <= 5; i++) {
    float gridX = map(i * 0.2, 0, 1, x + 40, x + width - 40);
    float gridY = map(i * 0.2, 0, 1, y + height - 40, y + 40);
    line(gridX, y + 40, gridX, y + height - 40); // Vertical grid line
    line(x + 40, gridY, x + width - 40, gridY); // Horizontal grid line
  }
}
```



What does that tells us 👍

- Population Density by Area: The plot shows a negative correlation between the area of a governorate and its population density. This suggests that as the area of a governorate increases, its population density tends to decrease. In other words, larger governorates, in terms of geographical area, tend to have fewer people living per square kilometer.
- Population by Area: This plot does not show a clear pattern or correlation between the area of a governorate and its population. This suggests that the size of a governorate (in terms of geographical area) does not necessarily determine its total population.
- Density by Population: The plot shows a positive correlation between the population of a governorate and its population density. This suggests that as the population of a governorate increases, its population density also tends to increase. In other words, governorates with larger populations tend to have more people living per square kilometer.